Docket No. AUS920031060US1 *PATENT*

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **Arndt et al.**	§	
	§	Group Art Unit: **2163**
Serial No. **10/777,724**	§	
	§	Examiner: **Black, Linh**
Filed: **February 12, 2004**	§	
	§	
For: **Architecture and Method for Managing the Sharing of Logical Resources Among Separate Partitions of a Logically Partitioned Computer System**	§	

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

35525
PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

APPEAL BRIEF (37 C.F.R. 41.37)

This brief is in furtherance of the Notice of Appeal, filed in this case on October 24, 2007.

A fee of $510.00 is required for filing an Appeal Brief. Please charge this fee to IBM Corporation Deposit Account No. 09-0447. No additional fees are believed to be necessary. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to IBM Corporation Deposit Account No. 09-0447. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to IBM Corporation Deposit Account No. 09-0447.

REAL PARTY IN INTEREST

The real party in interest in this appeal is the following party: International Business Machines Corporation of Armonk, New York.

RELATED APPEALS AND INTERFERENCES

This appeal has no related proceedings or interferences.

STATUS OF CLAIMS

A. TOTAL NUMBER OF CLAIMS IN APPLICATION

The claims in the application are: 1-21

B. STATUS OF ALL THE CLAIMS IN APPLICATION

Claims canceled: None

Claims withdrawn from consideration but not canceled: None

Claims pending: 1-21

Claims allowed: None

Claims rejected: 1-21

Claims objected to: None

C. CLAIMS ON APPEAL

The claims on appeal are: 1-21

STATUS OF AMENDMENTS

No amendments were submitted after the Final Office Action of September 21, 2007.

SUMMARY OF CLAIMED SUBJECT MATTER

A. CLAIM 1 - INDEPENDENT

The subject matter of claim 1 is directed to a method for managing shared resources in a logical partitioned data processing system. The method includes granting a logical resource (Specification page 15, line 5 – page 17, line 17; Figure 4, 402) from a server partition in the logical partitioned data processing system (Specification page 17, lines 7-17; Figure 4, 410) to a client partition in the logical partitioned data processing system (Specification page 15, lines 7-14; Figure 4, 420). The method further includes communicating an identifier (Specification page 16, line 23 – page 17, line 6; Figure 5, 504) from the server partition to the client partition (Specification page 17, lines 7-17; Figure 5, 504). The method further includes the client partition mapping the logical resource into a logical address space of the client partition (Specification page 16, line 23 – page 17, line 17; page 22, lines 8-10; Figure 5, 508, 510). The mapping is performed by the client partition responsive to the client partition accepting the identifier (Specification page 21, line 24 – page 22, line 11; Figure 5, 508, 510).

B. CLAIM 11 - INDEPENDENT

The subject matter of claim 11 is directed to a logical partitioned data processing system. The data processing system includes a server partition, at least one client partition, and a hypervisor (Specification, page 15, lines 5 – page 18, line 20; Figure 4). The hypervisor performs functions and services for partitions to create and enforce partitioning of the logical partitioned data processing system (Specification p. 12, lines 11-26). The server partition grants the client partition access to a logical resource (Specification page 15, line 5 – page 17, line 17; Figure 5, 502). The server partition communicates an identifier to the at least one client partition (Specification page 16, line 23 – page 17, line 17; Figure 5, 507). In response to accepting the identifier, a sharing client partition within the at least one client partition maps the logical resource into a logical address space of the sharing client partition (Specification page 16, line 23 – page 17, line 17; page 22, lines 8-10; Figure 5, 508, 510).

C. CLAIM 21 - INDEPENDENT

The subject matter of claim 21 is directed to a computer program product for managing shared resources in a logical partitioned data processing system. The computer program product includes instructions for granting a logical resource from a server partition in the logical partitioned data processing system to a client partition in the logical partitioned data processing system (Specification page 15, line 5 – page 17, line 17; Figure 4). The computer program product further includes instructions for the client partition to map the logical resource into a logical address space of the client partition (Specification page 16, line 23 – page 17, line 17; page 22, lines 8-10; Figure 5, 508, 510). The mapping is performed by the client partition in response to the client partition accepting the identifier (Specification page 21, line 24 – page 22, line 11; Figure 5, 508, 510).

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

The grounds of rejection to review on appeal are as follows:

A. GROUND OF REJECTION 1

Whether claims 1-21 are non-obvious over *Armstrong et al.,* Processor Reset Generated Via Memory Access Interrupt, U.S. Patent No. 6,467,007 (October 15, 2002) (hereinafter "*Armstrong*") in view of *Greene et al.,* Method and System for Managing Partitioned Data Resources, U.S. Patent No. 6,922,685 (July 26. 2005) (hereinafter "*Greene*").

ARGUMENT

A. GROUND OF REJECTION 1: Claims 1-21

The Examiner rejected claims 1-21 under 35 U.S.C. §103(a) as being obvious over *Armstrong* in view of *Greene*. Appellants request that the Board of Patent Appeals and Interferences overturn this clearly erroneous rejection.

A.1. A *Prima Facie* Case of Obviousness Has Not Been Established Against Claims 1-21

Claim 1 is representative of this claim grouping. Claim 1 is as follows:

> 1. A method for managing shared resources in a logical partitioned data processing system, the method comprising:
>
> granting, by a server partition in the logical partitioned data processing system, a logical resource to a client partition in the logical partitioned data processing system;
>
> communicating an identifier from the server partition to the client partition; and
>
> responsive to the client partition accepting the identifier, mapping the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition.

Regarding claim 1, the Examiner states the following:

> As per claim 1, *Armstrong* et al. teach logical partitioning with various resources in the physical computer – co. 1, lines 43-67; apparatus or computer 10 represents any of a number of multi-user computer systems such as a network server – col. 4, lines 9-23; a primary partition shares some of the partition management functions for the computer, such as handling the power on or powering off of the secondary partitions on computer 10 – col. 4, lines 55-67, thus, the primary partition acts as a server partition, and the secondary partitions are equivalent to the client partitions. *Armstrong* et al. teach the allocating/granting of resources – col. 5, lines 25-65; address translation tables/hardware page tables 90-94 are provided in the partition manager to respectively handle the virtual to real address translation/mapping operations; logical partition and resources can be in any combination and arrangement – col. 5, lines 23-65.
>
> However, *Armstrong* et al. do not suggest communicating an identifier from the server partition to the client partition and mapping the logical resources into a logical address space of the client partition, where the mapping is performed by the client partition. *Greene* et al. teach method and system for managing partitioned data resources – the title; clients' partitioning – col. 66, last paragraph, to col. 67, line 32; mapping of primary

> keys onto partition identifiers... to determine which partition contains the entity having a given PK – col. 68, line 10 to last paragraph; a central steward assigned for each entity type provides coordination and management of unique primary keys across all partitions – col. 63, lines 5-67. Thus it would have been obvious to one of ordinary skill in the art at the time of the invention to combine *Armstrong* et al.'s teaching with *Greene* et al.'s teaching to efficiently manage resources.

Final Office Action dated September 21, 2007, pp. 2-3.

The prior art reference (or references when combined) must teach or suggest all the claim limitations. *In re Royka,* 490 F.2d 981, 180 USPQ 580 (CCPA 1974). In determining obviousness, the scope and content of the prior art are... determined; differences between the prior art and the claims at issue are... ascertained; and the level of ordinary skill in the pertinent art resolved. Against this background the obviousness or non-obviousness of the subject matter is determined. *Graham v. John Deere Co.*, 383 U.S. 1 (1966). "Often, it will be necessary for a court to look to interrelated teachings of multiple patents; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by a person having ordinary skill in the art, all in order to determine whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue." *KSR Int'l. Co. v. Teleflex, Inc.*, No. 04-1350 (U.S. Apr. 30, 2007). "*Rejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness.*" Id. (citing *In re Kahn*, 441 F.3d 977, 988 (CA Fed. 2006)) (emphasis supplied).

Armstrong is directed to a multiprocessor environment where one processor can initiate the resetting of another processor. When a target processor "hangs" or encounters a partial failure, a service processor sets a flag at a static memory location in a local storage for the target processor. The service processor then invalidates each entry in an address translation table allocated to the target processor. Once each entry in the address table has been invalidated, the target processor regains control of its resources, and enters its known initial state. *See Armstrong*, col. 7, ll. 5-48.

Armstrong does not teach the feature in claim 1 of, "communicating an identifier from the server partition to the client partition, and responsive to the client partition accepting the identifier, mapping the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition." In addition, *Armstrong* teaches away from

"communicating an identifier" between server and client, as there is no stated need or provision for such communication evident in *Armstrong*.

Furthermore, *Greene* does not overcome the stated deficiencies of *Armstrong*. *Greene* is directed to a system of connecting remotely located separate server systems and databases across a network. The operating system of each of the remotely located server systems executes a Java virtual machine. Other server systems and databases are referenced from the Java virtual machine by using a virtual machine container. *Greene*'s "virtual machines" are Java virtual machines running on separate processing systems. *Greene*'s "partitions" are geographically disparate locations, or physically separate data processing systems. Thus, when *Greene* refers to the "partitioning" of resources, *Greene* is referring to a geographical grouping of servers containing information that is most often used by facilities at that geographical location. Thus, the portions of *Greene* cited by the examiner are not equivalent to the claimed features.

Still further, while the Examiner has cited several sections of *Greene* which detail the role of *Greene*'s entity manager, the Examiner has not pointed to any corresponding features of claim 1. Appellants therefore have no way to know with certainty which features of claim 1 to which the Examiner is referring. Therefore, the appellants deal with the possible correlations sequentially.

A.1.a. If the Examiner Equates *Greene*'s Steward to Appellants' Server Partition, then *Greene* Does Not Teach "Communicating an Identifier from the Server Partition to the Client Partition"

As stated above, the Examiner has not correlated any features of *Greene* to the features of claim 1. Rather the Examiner simply states the teaching of *Greene*, and expects the Appellants to supply the requisite analysis.

However, *Greene* does not teach, "communicating an identifier from the server partition to the client partition" when *Greene*'s steward is equated to Appellants' server partition. *Greene*'s steward 2010 issues a set of primary keys to satellites 2012-2018. The satellites 2012-2018 then assign the primary keys to a new instance when that instance is created. If the Examiner is equating satellites 2012-2018 with the Appellants' server partition, and steward 2010 with Appellants' client partition, such a correlation would impart a meaning to the claim terms that goes

beyond "their broadest reasonable interpretation consistent with the specification" under the standards of *Phillips v. AWH Corp.*, 415 F.3d 1303, 75 USPQ2d 1321 (Fed. Cir. 2005).

With regard to the Appellants' claimed feature of, "communicating an identifier from the server partition to the client partition," the Examiner appears to cite the following:

> Roles of the Entity Manger
>
> One of the main roles of this central manager is to provide coordination and management of unique primary keys (PKs) across all partitions. In the present architecture, all entities follow the convention of defining a candidate primary key consisting of a unique 64-bit integer called the UID (unique identifier). This UID provides a convenient foreign key that is used by externalized association engines to store references to entity instances, as will be further described below. In accordance with one exemplary embodiment of the present invention, one of the primary responsibilities of central entity manager 2010 is to maintain a block-up counter for generating new UIDs when a new block of primary keys is called for by any of satellites 2012-2018. Satellites 2012-2018 actually issue a primary key whenever an entity instance is created and not steward 2010. This approach avoids the necessity of accessing the manager upon every creation of a new entity instance. The satellite only need consult the steward during entity creation in the event that the satellite runs out of keys in its allocated block of keys. It must then go back to the steward to request another block of keys. This approach avoids the necessity of accessing the manager upon every creation of a new entity instance. The satellite only need consult the steward during entity creation in the even that the satellite runs out of keys. In accordance with another exemplary embodiment of the present invention, steward 2010 validates that a primary key proposed by a user for a new instance is not already in use by an existing instance. This latter sort of PK contrasts with the block-up UID generated by the central manager in that its form is dictated by the type of business object it represents. For example, the PK for a given entity might be a string or an integer, or it might be a composite key having more than one component. These domain-specific PKs would often be proposed by the application, or by custom logic within the entity implementation, and checked for uniqueness by the central entity manager, using for example, a hashing or directory service.
>
> In accordance with another exemplary embodiment of the present invention, steward 2010 serves as a place to keep the master data for the mapping of primary keys onto partition identifiers that indicate where each given object is stored. This is an alternative embodiment and is discussed more below with respect to multi-stage finders. However, in that case, when a cache miss is suffered out at a satellite server, the finder service faults over to the master data managed by steward 2010 to determine which partition contains the entity having a given PK. When a new entity is created, steward 2010 places a new entry in its master copy of the PK-to-partition map. This role of steward 2010, as the master record for this mapping, assumes that the multi-hop finder is based on distributed caches. If, as is discussed with respect to another embodiment of the present invention, enterprise repository 2030 is used for storing PK-to-partition maps, then burden for this data management shifts entirely

to enterprise repository 2030. However, if, and only if, the embodiment requires steward 2010 to generate new primary keys when new instances are created, and its

responsibility for recording the PK-to-partition association, then the central logic of the home interface's create operation can also be located within steward 2010.

Greene, col. 68, l. 10-col. 69, l. 3.

Steward 2010 of *Greene* is not a server partition as claimed in Appellants' claim 1 and defined in the specification. The steward is more akin to a hypervisor, or other partition management system, rather than a partition itself. Specifically with regard to steward 2010, *Greene* states the following:

> Steward 2010 provides a measure of central management to the present invention. With the storage and container servers for each entity being partitioned and distributed across physically separate server hosts, as described directly above, there is a need for a central manger, one for each entity class. The entity manager serves as a central authority for those aspects of the entity needing to be centralized, as represented in FIG. 20 as steward 2010.

Greene, col. 67, l. 61-col. 68, l. 2.

> [O]ne of the primary responsibilities of central entity manager 2010 is to maintain a block-up counter for generating new UIDs when a new block of primary keys is called for by any of satellites 2012-2018.

Greene, col. 68, ll. 20-23.

> [S]teward 2010 serves as a place to keep the master data for the mapping of primary keys onto partition identifiers that indicate where each given object is stored.

Greene col. 68, ll. 49-51.

> Finally, steward 2010 is responsible for finding an instance's partition container if the guidance stage of the find operation fails.

Greene col. 69, ll. 4-6.

Conversely, as defined by the Appellants, a partition, and specifically a client partition, is as follows:

> A logical partitioned functionality within a data processing system allows multiple copies of a single operating system or multiple heterogeneous operating systems to be simultaneously run on a single data processing system platform. A partition, within which an operating system image runs, is assigned a non-overlapping subset of the platform's physical resources. These platform allocable

> resources include one or more architecturally distinct processors with their interrupt management area, regions of system memory, and input/output (I/O) adapter bus slots. The partition's resources are represented by the platform's firmware to the operating system image.

Application as filed, pp. 1-2.

> A client partition is a partition with which the resource owner is authorized to share resources by its partition definition.

Application as filed, p. 15.

Therefore, Appellants' client partition is not the steward of *Greene*. The steward of the *Greene* and the client partition of the Appellants are two completely separate concepts. *Greene* therefore does not disclose or suggest, "communicating an identifier from the server partition to the client partition," as required by claim 1. *Armstrong* does not teach or suggest this claimed feature. Therefore, the proposed combination of references, considered as a whole, does not teach or suggest this claim feature. Therefore, under the standards of *In re Royka*, the examiner failed to state a *prima facie* obviousness rejection against the claims.

A.1.a.i. *Greene* Does Not Teach Mapping the Logical Resource into a Logical Address Space of the Client Partition, "*wherein the mapping is performed by the client partition*"

If the Examiner has equated *Greene*'s steward to Appellants' server partition, *Greene* does not teach mapping the logical resource into a logical address space of the client partition, "*wherein the mapping is performed by the client partition.*" *Greene* states the following:

> In accordance with one exemplary embodiment of the present invention, one of the primary responsibilities of central entity manager 2010 is to maintain a block-up counter for generating new UIDs when a new block of primary keys is called for by any of satellites 2012-2018. Satellites 2012-2018 actually issue a primary key whenever an entity instance is created and not steward 2010. This approach avoids the necessity of accessing the manager upon every creation of a new entity instance. The satellite only need consult the steward during entity creation in the event that the satellite runs out of keys in its allocated block of keys. It must then go back to the steward to request another block of keys. This approach avoids the necessity of accessing the manager upon every creation of a new entity instance. The satellite only need consult the steward during entity creation in the even that the satellite runs out of keys.

Greene, col. 68, ll. 24-34.

Thus, while the *Greene* states that the steward provides a set of allocated keys to the satellite, the cited section of *Greene* is silent as to which of those entities actually performs a mapping function. Because the keys are transferred as a group and not assigned to the new entity when the keys are transferred, it appears that no mapping can occur until the satellite actually assigns the allocated key to the subsequently created new entity. However, the cited section of *Greene* appears silent in this regard, and the Examiner has not provided any explanation of the interpreted mapping. *Armstrong* does not teach or suggest this claimed feature. Therefore, the proposed combination of references, considered as a whole, does not teach or suggest this claim feature. Therefore, under the standards of *In re Royka*, the examiner failed to state a *prima facie* obviousness rejection against the claims.

A.1.b. If the Examiner equates *Greene*'s steward to Appellants' Server Partition, then *Greene* Does Not Teach "communicating an identifier from the *server partition* to the *client partition*"

Because of the opacity of the Examiner's rejection, the Appellants also address the possibility of the Examiner equating the Appellants' client partition to *Greene*'s satellite. The steward of *Greene* is defined and discussed in section A.1.a above.

Greene's steward is more akin to a hypervisor, or other partition management system, rather than a partition itself. The steward maintains a block-up counter for generating new user identifications when a new block of primary keys is called for by any of the satellites. The steward serves as a place to keep the master data for the mapping of primary keys onto partition identifiers that indicate where each given object is stored. Finally, the steward is responsible for finding an instance's partition container if the guidance stage of the find operation fails.

Conversely, as defined by the Appellants, a partition, and specifically a server partition, is as follows:

> A logical partitioned functionality within a data processing system allows multiple copies of a single operating system or multiple heterogeneous operating systems to be simultaneously run on a single data processing system platform. A partition, within which an operating system image runs, is assigned a non-overlapping subset of the platform's physical resources. These platform allocable resources include one or more architecturally distinct processors with their interrupt management area, regions of system memory, and input/output (I/O) adapter bus

slots. The partition's resources are represented by the platform's firmware to the operating system image.

Application as filed, pp. 1-2.

> [T]he server partition performs a grant operation specifying a resource to be shared... [with] a client partition.

Application as filed, p. 21.

The steward of the *Greene* and the server partition of the Appellants are two completely separate concepts. *Greene*'s steward provides central management and organization for the distributed network. The Appellants' server partition is itself a partitioned functionality within the data processing system that allows an instance of an operating system to be run thereon.

Appellants' server partition is not the steward of *Greene*. *Greene* therefore does not disclose or suggest, "communicating an identifier from the server partition to the client partition," as required by claim 1. *Armstrong* does not teach or suggest this claimed feature. Therefore, the proposed combination of references, considered as a whole, does not teach or suggest this claim feature. Therefore, under the standards of *In re Royka*, the examiner failed to state a *prima facie* obviousness rejection against the claims.

A.1.c. *Greene* Teaches Away from Mapping the Logical Resource "*responsive to the client partition accepting the identifier*"

A reference may be said to "teach away" from the claimed invention when a person of ordinary skill, upon reading the reference, would be discouraged from following the path set out in the reference, or would be led in a direction divergent from the path that was taken by the Appellants. *In re Gurley*, 27 F.3d 551, 553, 31 U.S.P.Q.2D 1130, 1131 (Fed. Cir. 1995).

With regard to mapping the primary keys, *Greene* discloses:

> In accordance with one exemplary embodiment of the present invention, one of the primary responsibilities of central entity manager 2010 is to maintain a block-up counter for generating new UIDs when a new block of primary keys is called for by any of satellites 2012-2018. Satellites 2012-2018 actually issue a primary key whenever an entity instance is created and not steward 2010. This approach avoids the necessity of accessing the manager upon every creation of a new entity instance. The satellite only need consult the steward during entity creation in the event that the satellite runs out of keys in its allocated block of keys. It must then go back to the steward to request another block of keys. This approach

> avoids the necessity of accessing the manager upon every creation of a new entity instance. The satellite only need consult the steward during entity creation in the even that the satellite runs out of keys.

Greene, col. 68, ll. 24-34.

As can be seen from the above passage, if the Examiner has equated *Greene*'s primary key to the Appellants' identifier, the logical resource is not mapped "*responsive to the client partition accepting the identifier.*" Satellites 2012-2018 accept a group of keys from the steward. The group of keys is then retained at the satellite for future distribution to a newly created entity.

If the Examiner equates *Greene*'s primary key to Appellants' identifier, and *Greene*'s satellites to Appellants' client partitions, then the new entity is mapped responsive to creating the entity, not responsive to accepting the identifier, as required by claim 1. The satellites of *Greene* accept the group of identifiers, and then assign an identifier from the group when an entity is created. *Greene* does not teach that each time that an identifier is received, the logical resource is mapped. In fact, *Greene* specifically teaches away from mapping the logical resource "*responsive to the client partition accepting the identifier*" by stating:

> This approach avoids the necessity of accessing the manager upon every creation of a new entity instance. The satellite only need consult the steward during entity creation in the even that the satellite runs out of keys.

Greene, col. 68, ll. 30-34.

Thus *Greene* sets out distinct disadvantages of mapping the logical resource "*responsive to the client partition accepting the identifier*" which would discourage one of ordinary skill in the art from following the path set out in *Greene*. Any rejection based on *Greene* for the stated reasons is therefore improper. Furthermore, *Armstrong* does not teach or suggest this claimed feature. Therefore, the proposed combination of references, considered as a whole, does not teach or suggest this claim feature. Therefore, under the standards of *In re Royka*, the examiner failed to state a *prima facie* obviousness rejection against the claims.

A.1.d. *Greene* Does not Teach "mapping the *logical resource* into a *logical address space* of the client partition"

With regard to an entity instance, which the Examiner appears to equate the Appellants' logical address space, *Greene* states the following:

> The entity instances are simply Java objects which conform to some strict conventions and live in a container environment. The entity instances themselves will interact with the backing data store, typically via Structured Query Language (SQL) calls to a Relational DataBase Management System (RDBMS). However, all application interaction with data is mediated through the instances, thus applications do not directly interact with the data store level of the DataBus. All entity instances must offer both a home interface and an instance interface. The home interface supports class-level functions, such as creation and initialization of new entity instances; finders allow query for existing entity instances matching specified criteria; and methods for permanently deleting existing entities from persistent storage. Finally, the containers should have a sophisticated model for caching entity instances in-core, managing the life-cycle of cached instances as they move in and out of cache and are created and destroyed, management of concurrent access by multiple users, and management of security (access control) and transactions.

Greene, col. 62, ll. 28-47.

The entity instance is a local instance of information that is stored at a remote location. Responsive to a user initiation of the java container, the java container retrieves information from the remote location and runs a local instance thereof. The Examiner therefore seems to correlate this data, which is stored in the java container, to the Appellants' logical resource. With regard to the logical resource, the Appellants' state the following:

> A logical resource may include, without limitation, a processor, a region of main storage, an I/O adapter register, a platform interrupt, and the like.

Application as filed, p. 15.

Despite the inclusion of the statement "and the like," it is clear that the data stored in the entity instance of *Greene* is not a logical resource, as claimed by the Appellants. The Appellants' logical resource is a physical resource, or a machine interaction with such a physical resource. *Greene*'s data stored in the entity instance is a user-interactive interface for providing the user access to remotely stored information.

This data in *Greene*'s interactive entity instance is completely different from the physical resources and basic machine interactions that are the logical resources provided in claim 1. Therefore, *Greene* does not disclose "mapping the *logical resource* into a *logical address space* of the client partition" as recited in claim 1. Furthermore, *Armstrong* does not teach or suggest this claimed feature. Therefore, the proposed combination of references, considered as a whole, does

not teach or suggest this claim feature. Therefore, under the standards of *In re Royka*, the examiner failed to state a *prima facie* obviousness rejection against the claims.

A.2. No Reason to Combine or Alter the References Has Been Stated

A.2.a. No Articulated Reasoning with a Rational Underpinning Has Been Articulated to Support the Legal Conclusion of Obviousness

"*Rejections on obviousness grounds cannot be sustained by mere conclusory statements; instead, there must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness.*" *KSR Int'l. Co. v. Teleflex, Inc.*, No. 04-1350 (U.S. Apr. 30, 2007) (citing *In re Kahn*, 441 F.3d 977, 988 (CA Fed. 2006).

As pointed out above, the Examiner has cited voluminous sections of *Greene* without specifically pointing to any disclosure therein that equate to those features recited in Appellants' claim 1. The Examiner simply states the teaching of *Greene*, and expects the Appellants to supply the requisite analysis.

In the case at hand, the Examiner makes only a conclusory statement that provides no articulated reasoning and provides no rational underpinning to support the legal conclusion of obviousness. Instead, the Examiner only states that the combination would be obvious because the combination would "efficiently manage resources."

The Examiner's statement is conclusory because the Examiner provides no support for the proposition that the combination would actually accomplish what the Examiner purports the combination to accomplish. The Examiner's statement is conclusory also because the Examiner provides no technical argument or other basis that one of ordinary skill in the art would recognize the purported advantage or have a reason to implement the advantage. The Examiner's statement is conclusory also because the Examiner provides no technical argument or other basis that the advantage exists in the first place vis-à-vis the claimed invention.

Additionally, the Examiner's statement provides no rational underpinning to support the legal conclusion of obviousness. The Examiner only states a purported advantage to combine the references. However, the Examiner does not provide any technical or other rational connection between the purported advantage and the legal conclusion of obviousness. Instead the Examiner

simply states the purported advantage. The Examiner assumes that the Appellants would recognize that the purported advantage would somehow compel the legal conclusion that claim 1 is obvious in view of the references. Such an assumption does not comport with the requirements of *KSR Int'l.*

Therefore, the Examiner's statement is both conclusory and provides no articulated reasoning with some rational underpinning to support the legal conclusion of obviousness. Accordingly, under the standards of *KSR Int'l. Co. v. Teleflex, Inc.*, the Examiner failed to state a *prima facie* obviousness rejection against claim 1 or any other claim in this grouping of claims.

A.2.b. No Proper Reason Exists To Combine the References in a Manner that Compels the Legal Conclusion that Claim 1 Is Obvious in View of the References

Additionally, no proper reason exists to combine the references in a manner that compels the legal conclusion that claim 1 is obvious in view of the references. No proper reason to so combine the references exists because the references are completely different from each other. For example, *Armstrong* is directed towards a processor reset generated via memory access interrupt. *Armstrong*, title. In stark contrast, *Greene* is directed to the completely different field of managing remotely located separate server systems and databases across a network using a Java virtual machine container. In still further contrast to *Greene*, claim 1 is directed towards managing shared resources in a logical partitioned data processing system. Thus, *Greene* has nothing to do with either *Armstrong*, or Appellants' claim 1, except insomuch as all three use a data processing system for implementation.

Because the references have nothing to do with each other, and because *Greene* has nothing to do with claim 1, one of ordinary skill could find no reason to combine the references to achieve the invention of claim 1, when the references are considered together as a whole. More importantly, the vast disparities among the references show that one of ordinary skill *could not establish a rational reason to combine the references in a manner that compels the conclusion that claim 1 is obvious* in view of the references considered together as a whole. Accordingly, under the standards of *KSR Int'l.*, the Examiner failed to state a *prima facie* obviousness rejection against claim 1.

B. CONCLUSION

As shown above, the Examiner has failed to state valid rejections against any of the claims. Therefore, Appellants request that the Board of Patent Appeals and Interferences reverse the rejections.

/Brandon G. Williams/
Brandon G. Williams
Reg. No. 48,844
YEE & ASSOCIATES, P.C.
PO Box 802333
Dallas, TX 75380
(972) 385-8777

CLAIMS APPENDIX

The text of the claims involved in the appeal is as follows:

1. A method for managing shared resources in a logical partitioned data processing system, the method comprising:

granting, by a server partition in the logical partitioned data processing system, a logical resource to a client partition in the logical partitioned data processing system;

communicating an identifier from the server partition to the client partition; and

responsive to the client partition accepting the identifier, mapping the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition.

2. The method of claim 1, further comprising:

generating the identifier for the logical resource, wherein the identifier is generated by a hypervisor.

3. The method of claim 2, wherein the identifier is unique within the client partition.

4. The method of claim 2, wherein the identifier cannot be used to access the logical resource outside the client partition.

5. The method of claim 1, further comprising:

returning, by the client partition, the logical resource to the server partition.

6. The method of claim 5, further comprising:

rescinding, by the server partition, the logical resource.

7. The method of claim 1, further comprising:

responsive to a determination, at the server partition, that the client partition is incapable of gracefully returning the logical resource, performing a forced rescind operation.

8. The method of claim 7, further comprising:

preventing translation tables in the client partition from containing references to a physical address of the logical resource.

9. The method of claim 1, further comprising:

responsive to a failure of the server partition, notifying the client partition of the failure of the server partition;

recovering outstanding shared logical resources for the server partition; and

restarting the server partition.

10. The method of claim 9, further comprising: delaying for a period of time prior to the step of recovering the outstanding shared logical resources for the server partition.

11. A logical partitioned data processing system, comprising:

a server partition;

at least one client partition; and

a hypervisor, wherein the hypervisor performs functions and services for partitions to create and enforce partitioning of the logical partitioned data processing system, wherein the server partition grants access to a logical resource to the at least one client partition, wherein the server partition communicates an identifier to the at least one client partition, and wherein a sharing client partition, within the at least one client partition, maps the logical resource into a logical address space of the sharing client partition in response to accepting the identifier

12. The logical partitioned data processing system of claim 11, wherein the hypervisor generates the identifier for the logical resource.

13. The logical partitioned data processing system of claim 12, wherein the identifier is unique within the sharing client partition.

14. The logical partitioned data processing system of claim 12, wherein the identifier cannot be used to access the logical resource outside the sharing client partition.

15. The logical partitioned data processing system of claim 11, wherein the sharing client partition returns the logical resource to the server partition.

16. The logical partitioned data processing system of claim 15, wherein the server partition rescinds the logical resource.

17. The logical partitioned data processing system of claim 11, wherein the server partition, responsive to a determination that the sharing client partition is incapable of gracefully returning the logical resource, performs a forced rescind operation.

18. The logical partitioned data processing system of claim 17, wherein the hypervisor, responsive to the forced rescind operation, prevents translation tables in the sharing client partition from containing references to a physical address of the logical resource.

19. The logical partitioned data processing system of claim 11, wherein the hypervisor, responsive to a failure of the server partition, notifies the sharing client partition of the failure of the server partition, recovers outstanding shared logical resources for the server partition, and restarts the server partition.

20. The logical partitioned data processing system of claim 19, wherein the hypervisor delays for a period of time prior to recovering the outstanding shared logical resources for the server partition.

21. A computer program product for managing shared resources in a logical partitioned data processing system, the computer program product comprising:

instructions for granting, by a server partition in the logical partitioned data processing system, a logical resource to a client partition in the logical partitioned data processing system;

instructions for communicating an identifier from the server partition to the client partition; and

instructions, responsive to the client partition accepting the identifier, for mapping the logical resource into a logical address space of the client partition, wherein the mapping is performed by the client partition.

EVIDENCE APPENDIX

This appeal brief presents no additional evidence.

RELATED PROCEEDINGS APPENDIX

This appeal has no related proceedings.